

# Towards a Theory of Complexity of Regular Languages<sup>\*</sup>

Janusz A. Brzozowski<sup>1</sup>

David R. Cheriton School of Computer Science, University of Waterloo  
Waterloo, ON, Canada N2L 3G1  
brzozo@uwaterloo.ca

**Abstract.** We survey recent results concerning the complexity of regular languages represented by their minimal deterministic finite automata. In addition to the quotient complexity of the language – which is the number of its (left) quotients, and is the same as its state complexity – we also consider the size of its syntactic semigroup and the quotient complexity of its atoms – basic components of every regular language. We then turn to the study of the quotient/state complexity of common operations on regular languages: reversal, (Kleene) star, product (concatenation) and boolean operations. We examine relations among these complexity measures. We discuss several subclasses of regular languages defined by convexity. In many, but not all, cases there exist “most complex” languages, languages satisfying all these complexity measures.

**Keywords:** atom, boolean operation, complexity measure, concatenation, convex language, most complex language, quotient complexity, regular language, reversal, star, state complexity, syntactic semigroup, unrestricted complexity

## 1 Introduction

We assume the reader is familiar with basic properties of regular languages and finite automata, as discussed in [55,62], for example; formal definitions are given later.

We study the complexity of regular languages represented by their minimal deterministic finite automata (DFAs). The number of states in the minimal DFA of a language is its *state complexity* [51,63]; this number is used as a first measure of complexity. But languages having the same state complexity can be quite simple or very complex. How do we decide whether one language is more complex than another? In this respect, the size of the syntactic semigroup of the language – which is isomorphic to the transition semigroup of its minimal DFA – appears to be a good measure.

Another way to distinguish two regular languages of the same state complexity is by comparing how difficult it is to perform operations on these languages.

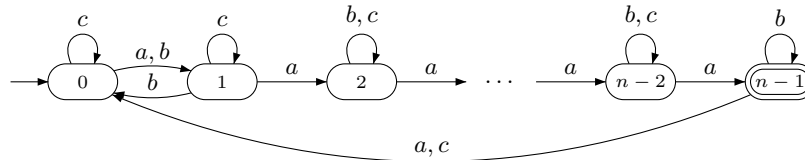
---

<sup>\*</sup> This work was supported by the Natural Sciences and Engineering Research Council of Canada grant No. OGP0000871.

The state complexity of a regularity preserving unary operation on a language is defined as the maximal complexity of the result of the operation expressed as a function of the state complexity of the language. For example, we know that there are regular languages of state complexity  $n$  whose reverses have state complexity  $2^n$ , but many languages do not meet this bound. For binary operations we have two languages of state complexities  $m$  and  $n$ , respectively. The state complexity of a binary operation is the maximal state complexity of the result, expressed as a function of  $m$  and  $n$ .

In general, to establish the state complexity of a unary operation, we need to find an upper bound on this complexity and a language for each  $n$  that meets this bound. This sequence of languages is called a *stream*. The languages in the stream often have the same structure and differ only in the parameter  $n$ . For binary operations we need two streams. For some operations the same stream can be used for both operands. However, if the second operand cannot be the same as the first, it can usually be a *dialect* of the first operand – a language that differs only slightly from the first.

It has been proved [8] that the stream  $(L_3(a, b, c), \dots, L_n(a, b, c), \dots)$  of regular languages shown in Fig. 1 is *most complex* because it meets the following complexity bounds: the size of the syntactic semigroup, and the state complexities reversal, (Kleene) star, product/concatenation, and all binary boolean operations. It also has the largest number  $2^n$  of atoms (discussed later), and all the atoms have maximal state complexity.



**Fig. 1.** Minimal DFA of a most complex regular language  $L_n(a, b, c)$ .

The *alphabet* of a regular language  $L$  is  $\Sigma$  (or  $L$  is a language over  $\Sigma$ ) if  $L \subseteq \Sigma^*$  and every letter of  $\Sigma$  appears in a word of  $L$ . In addition to the usual state complexity of binary operations on languages over the same alphabet, *unrestricted* state complexity on languages over different alphabets has also been studied [10]. By adding an input  $d$  that induces the identity transformation in the DFA of Fig. 1, we obtain a most complex language that also meets the bounds for unrestricted operations.

A natural question then arises whether most complex language streams also exist in proper subclasses of regular languages. The answer is positive for many, but not all, classes. A rich source of subclasses is provided by the concept of

convexity. In this paper we summarize the results for many classes of convex languages.

Many of these results were presented as an invited talk at the *20th International Conference on Developments in Language Theory*, Montréal, Québec on July 25, 2016. A short abstract appeared in [9].

## 2 Quotient/State Complexity of Regular Languages

Let  $\Sigma = \{a_1, \dots, a_k\}$  be a nonempty set, called an *alphabet*, consisting of *letters*  $a_i$ ,  $i = 1, \dots, k$ . A *word* over  $\Sigma$  is a sequence  $a_{i_1} \dots a_{i_m}$ , where  $a_{i_j} \in \Sigma$ ,  $j = 1, \dots, m$ ; if  $m = 0$ , the word is *empty* and is denoted by  $\varepsilon$ . A *language* over  $\Sigma$  is any subset of  $\Sigma^*$ , where  $\Sigma^*$  is the free monoid generated by  $\Sigma$  with  $\varepsilon$  as the identity, that is,  $\Sigma^*$  is the set of all words over  $\Sigma$ . Recall that if  $L$  is a language over  $\Sigma$ , every letter of  $\Sigma$  appears in at least one word of  $L$ .

The languages  $\emptyset$  (the empty language) and  $\{a_i\}$ ,  $i = 1, \dots, k$  (the *letter* languages) are called *basic*. A language is *regular* if it can be constructed from the basic languages using only the operations union (denoted by  $L \cup L'$ ), product (concatenation) (denoted by juxtaposition:  $LL' = \{w \mid w = xy, x \in L, y \in L'\}$ ), and star (denoted by  $L^* = \bigcup_{n \geq 0} L^n$ , where  $L^0 = \{\varepsilon\}$ , and  $L^{n+1} = L^n L$ ).

If  $w \in \Sigma^*$  and  $L \subseteq \Sigma^*$ , the (*left*) *quotient* of  $L$  by  $w$  is the language  $w^{-1}L = \{x \mid wx \in L\}$ ; it is the set of “all words that can follow  $w$  in  $L$ ”. It is well known that a language is regular if and only if it has a finite number of distinct quotients [6,54]. So it is natural to consider the number of quotients of a regular language  $L$  as a complexity measure, which we call the *quotient complexity* of  $L$  and denote by  $\kappa(L)$ .

Quotients can be computed as follows: For  $a, b \in \Sigma$ ,  $w \in \Sigma^*$  and  $L \subseteq \Sigma^*$  we have

$$a^{-1}L = \begin{cases} \emptyset, & \text{if } L = \emptyset, \text{ or } L = \{b\} \text{ and } a \neq b; \\ \{\varepsilon\}, & \text{if } L = a. \end{cases} \quad (1)$$

$$a^{-1}(L \cup L') = a^{-1}L \cup a^{-1}L'. \quad (2)$$

$$a^{-1}LL' = \begin{cases} (a^{-1}L)L', & \text{if } \varepsilon \notin L; \\ (a^{-1}L)L' \cup a^{-1}L', & \text{if } \varepsilon \in L. \end{cases} \quad (3)$$

$$a^{-1}(L^*) = (a^{-1}L)L^*. \quad (4)$$

$$\varepsilon^{-1}L = L. \quad (5)$$

$$(wa)^{-1}L = a^{-1}(w^{-1}L). \quad (6)$$

When we compute quotients this way, they are represented by expressions involving the basic languages, union, product and star, and it may not be obvious that two different expressions denote the same quotient. However, it is easy to recognize *similarity*, where two expressions are *similar* [6] if one can be obtained from the other using the following rules:

$$L \cup L = L, \quad L \cup L' = L' \cup L, \quad L \cup (L' \cup L'') = (L \cup L') \cup L'', \quad (7)$$

$$L \cup \emptyset = L, \quad \emptyset L = L\emptyset = \emptyset, \quad \{\varepsilon\}L = L\{\varepsilon\} = L. \quad (8)$$

The number of dissimilar expressions of a regular language is always finite [6].

A concept closely related to a regular language is that of a *deterministic finite automaton (DFA)*, which is a quintuple  $\mathcal{D} = (Q, \Sigma, \delta, q_0, F)$ , where  $Q$  is a finite non-empty set of *states*,  $\Sigma$  is a finite non-empty *alphabet*,  $\delta: Q \times \Sigma \rightarrow Q$  is the *transition function*,  $q_0 \in Q$  is the *initial state*, and  $F \subseteq Q$  is the set of *final states*. We extend  $\delta$  to functions  $\delta: Q \times \Sigma^* \rightarrow Q$  and  $\delta: 2^Q \times \Sigma^* \rightarrow 2^Q$  as usual. A DFA  $\mathcal{D}$  *accepts* a word  $w \in \Sigma^*$  if  $\delta(q_0, w) \in F$ . The set of all words accepted by  $\mathcal{D}$  is the *language* accepted by  $\mathcal{D}$ , denoted by  $L(\mathcal{D})$ . If  $q$  is a state of  $\mathcal{D}$ , then the language  $L_q(\mathcal{D})$  of  $q$  is the language accepted by the DFA  $(Q, \Sigma, \delta, q, F)$ . A state is *empty* if its language is empty. Two states  $p$  and  $q$  of  $\mathcal{D}$  are *equivalent* if  $L_p(\mathcal{D}) = L_q(\mathcal{D})$ . A state  $q$  is *reachable* if there exists  $w \in \Sigma^*$  such that  $\delta(q_0, w) = q$ . A DFA is *minimal* if all of its states are reachable and no two states are equivalent.

The famous theorem of Kleene [49] states that a language is regular if and only if it is accepted by a DFA. We can derive a DFA accepting a regular language  $L$  directly from its quotients. Denote the set of quotients of  $L$  by  $K = \{K_0, \dots, K_{n-1}\}$ , where  $K_0 = L = \varepsilon^{-1}L$  by convention. Each quotient  $K_i$  can be represented also as  $w_i^{-1}L$ , where  $w_i \in \Sigma^*$  is such that  $w_i^{-1}L = K_i$ . Now define the *quotient DFA* of  $L$  as follows:  $\mathcal{D} = (K, \Sigma, \delta, K_0, F)$ , where  $\delta(K_i, a) = K_j$  if  $a^{-1}K_i = K_j$ , and  $F = \{K_i \mid \varepsilon \in K_i\}$ . This DFA accepts  $L$  and is minimal<sup>1</sup>.

In any DFA  $\mathcal{D} = (Q, \Sigma, \delta, q_0, F)$ , if  $\delta(q_0, w) = q$ , then  $L_q(\mathcal{D}) = L(Q, \Sigma, \delta, q, F)$ , known also as the *right language of  $q$* , is precisely the quotient  $w^{-1}L$ . Evidently, the state complexity of a language is equal to its quotient complexity. From now on we refer to the quotient/state complexity of  $L$  simply as the *complexity* of  $L$ .

### 3 Syntactic/Transition Semigroups

According to our complexity measure any two languages with  $n$  quotients have the same complexity. But consider the language  $L_n$  accepted by the minimal DFA of Fig. 1 and the language  $L'_n = \Sigma^{n-2}$ . Intuitively  $L'_n$  is much simpler than  $L_n$ .

It was proposed in [8] that the size of the syntactic semigroup of a language should be used as an additional complexity measure. We proceed to define it now.

The *Myhill congruence*  $\approx_L$  [53], also known as the *syntactic congruence*, of a language  $L \subseteq \Sigma^*$  is defined on  $\Sigma^+$  as follows: For  $x, y \in \Sigma^+$ ,

$$x \approx_L y \text{ if and only if } wxz \in L \Leftrightarrow wyz \in L, \text{ for all } w, z \in \Sigma^*.$$

<sup>1</sup> If a DFA is constructed using dissimilar expressions and is not minimal, it can be minimized by one of several methods [5,45,52], by merging states corresponding to the same expression.

The quotient set  $\Sigma^+/\approx_L$  of equivalence classes of  $\approx_L$  is a semigroup, the *syntactic semigroup*  $T_L$  of  $L$ . The *syntactic complexity* of a language  $L$  is the cardinality of the syntactic semigroup.

Returning to our example, the syntactic complexity of  $L_n$  is known to be  $n^n$ , whereas that of  $L'_n = \Sigma^{n-2}$  is  $n-1$ ; hence syntactic complexity clearly distinguishes the two languages.

Let  $Q_n$  be a set of  $n$  elements. Without loss of generality, we assume  $Q_n = \{0, 1, \dots, n-1\}$ . A *transformation* of  $Q_n$  is a mapping  $t: Q_n \rightarrow Q_n$ . The *image* of  $q \in Q_n$  under  $t$  is denoted by  $qt$ . If  $s, t$  are transformations of  $Q_n$ , their composition is defined by  $q(st) = (qs)t$ . Let  $\mathcal{T}_{Q_n}$  be the set of all  $n^n$  transformations of  $Q_n$ ; then  $\mathcal{T}_{Q_n}$  is a monoid under composition.

For  $k \geq 2$ , a transformation  $t$  of a set  $P = \{q_0, q_1, \dots, q_{k-1}\} \subseteq Q_n$  is a *k-cycle* if  $q_0t = q_1, q_1t = q_2, \dots, q_{k-2}t = q_{k-1}, q_{k-1}t = q_0$ . This *k-cycle* is denoted by  $(q_0, q_1, \dots, q_{k-1})$ , and it acts as the identity on the states not in the cycle. A 2-cycle  $(q_0, q_1)$  is a *transposition*. A transformation that sends all the states of  $P$  to  $q$  and acts as the identity on the remaining states is denoted by  $(P \rightarrow q)$ . If  $P = \{p\}$  we write  $(p \rightarrow q)$  for  $(\{p\} \rightarrow q)$ . The identity transformation is denoted by  $\mathbb{1}$ . The notation  $(\overset{j}{i} q \rightarrow q+1)$  denotes a transformation that sends  $q$  to  $q+1$  for  $i \leq q \leq j$  and is the identity for the remaining states, and  $(\overset{j}{i} q \rightarrow q-1)$  is defined similarly.

Let  $\mathcal{D} = (Q_n, \Sigma, \delta, q_0, F)$  be a DFA, where we use  $Q_n = \{0, \dots, n-1\}$  as the set of states, without loss of generality. Each word  $w \in \Sigma^+$  induces a transformation  $\delta_w$  of the set  $Q_n$  defined by  $q\delta_w = \delta(q, w)$ ; we denote this by  $w: \delta_w$ . Sometimes we use the word  $w$  to denote the transformation it induces; thus we write  $qw$  instead of  $q\delta_w$ . We extend the notation to sets of states: if  $P \subseteq Q_n$ , then  $Pw = \{pw \mid p \in P\}$ . We also write  $P \xrightarrow{w} Pw$  to mean that the image of  $P$  under  $w$  is  $Pw$ .

The set  $T_{\mathcal{D}}$  of all transformations induced by non-empty words forms a semigroup of transformations called the *transition semigroup* of  $\mathcal{D}$  [56]. This semigroup is generated by  $\{\delta_a \mid a \in \Sigma\}$ . We use the transition semigroup rather than the transition monoid, because the latter always has the identity transformation induced by the empty word, whereas, in the semigroup, if the identity exists it must be induced by a non-empty word. For a more detailed discussion of the necessity of distinguishing between semigroups and monoids see [39, Chapter V], for example.

If  $\mathcal{D}_n$  is a minimal DFA of  $L_n$ , then  $T_{\mathcal{D}_n}$  is isomorphic to the syntactic semigroup  $T_{L_n}$  of  $L_n$  [56], and we represent elements of  $T_{L_n}$  by transformations in  $T_{\mathcal{D}_n}$ . We return to syntactic complexity later.

## 4 Quotients

Since quotients play a key role in defining a regular language we should also consider their complexity. In our example of Fig. 1 all quotients have complexity  $n$ . In the case of the language  $L'_n = \Sigma^{n-2}$ , the quotients  $\varepsilon^{-1}L'_n, a^{-1}L'_n, \dots, a^{n-2}L'_n, a^{n-1}L'_n$ , where  $a \in \Sigma$  have complexities  $n, n-1, \dots, 2, 1$ , respectively.

In general, however, the complexity of quotients is not a very good measure because it is always  $n$  if the DFA is strongly connected. But to ensure that most complex languages also have most complex quotients, we add the complexities of quotients as one of our measures.

## 5 Atoms

Atoms of regular languages were introduced in [34] as intersections of quotients. Atoms as congruence classes were presented in [46]. Quotient complexities of atoms were studied in [33,46].

For a regular language  $L$  and words  $x, y \in \Sigma^*$  consider the left congruence:

$$x \triangleleft_L y \text{ if and only if } ux \in L \Leftrightarrow uy \in L \text{ for all } u \in \Sigma^*.$$

An *atom* is a congruence class of  $\triangleleft_L$ ; thus two words  $x$  and  $y$  are in the same class if  $x \in u^{-1}L \Leftrightarrow y \in u^{-1}L$  for all  $u \in \Sigma^*$ . If  $Q_n = \{0, \dots, n-1\}$  and  $L$  is a regular language with quotients  $K = \{K_0, \dots, K_{n-1}\}$ , then each subset  $S$  of  $Q_n$  defines an *atomic intersection*  $A_S = \bigcap_{i \in S} K_i \cap \bigcap_{i \in \overline{S}} \overline{K_i}$ , where  $\overline{S} = Q_n \setminus S$  and  $\overline{L} = \Sigma^* \setminus L$  for any  $L \subseteq \Sigma^*$ ; an atom of  $L$  is a non-empty atomic intersection. It follows that each quotient  $K_i$  is a union of atoms, namely of all the atoms in which  $K_i$  appears uncomplemented. It is also known that quotients of atoms are unions of atoms [34]. Thus atoms are fundamental components of a language, and it was proposed in [8] that the quotient complexity of atoms should be considered as a complexity measure of regular languages.

A *nondeterministic finite automaton (NFA)* is a quintuple  $\mathcal{N} = (Q, \Sigma, \delta, I, F)$ , where  $Q$ ,  $\Sigma$  and  $F$  are as in a DFA,  $\delta: Q \times \Sigma \rightarrow 2^Q$ , and  $I \subseteq Q$  is the *set of initial states*. Each triple  $(p, a, q)$  with  $p, q \in Q$ ,  $a \in \Sigma$  is a *transition* if  $q \in \delta(p, a)$ . A sequence  $((p_0, a_0, q_0), (p_1, a_1, q_1), \dots, (p_{k-1}, a_{k-1}, q_{k-1}))$  of transitions, where  $p_{i+1} = q_i$  for  $i = 0, \dots, k-2$  is a *path* in  $\mathcal{N}$ . The word  $a_0a_1 \dots a_{k-1}$  is the word *spelled* by the path. A word  $w$  is accepted by  $\mathcal{N}$  if there exists a path with  $p_0 \in I$  and  $q_{k-1} \in F$  that spells  $w$ .

Recall that we have defined the quotient DFA of a regular language  $L$  using its quotients as states. In an analogous way, we define an NFA called the *átomaton*<sup>2</sup> of  $L$  using atoms as states. The átomaton of  $L$  is a NFA  $\mathcal{A} = (A, \Sigma, \alpha, I_{\mathcal{A}}, \{A_{p-1}\})$ , where  $A$  is the set of atoms of  $L$ ;  $\alpha$  is the transition function defined by  $A_j \in \alpha(A_i, a)$  if  $aA_j \subseteq A_i$ ;  $I_{\mathcal{A}}$  is the set of initial atoms, those atoms in which  $L = K_0$  appears uncomplemented; and  $A_{p-1}$  is the final atom: the only atom containing  $\varepsilon$ . In the átomaton, the right language of state  $A_i$  is the atom  $A_i$ .

We denote by  $L^R$  the reverse of the language  $L$ . Let  $\mathbb{R}$  be the NFA operation that interchanges the sets of initial and final states and reverses all transitions. Let  $\mathbb{D}$  be the NFA operation that determinizes a given NFA using the subset construction and taking into account only the subsets reachable from the set

<sup>2</sup> The accent is added to indicate that the word should be pronounced with the stress on the first syllable, and also to avoid confusion between *automaton* and *atomaton*.

of initial states. Finally, let  $\mathbb{M}$  be the minimization operation of DFAs. These operations are applied from left to right; thus in  $\mathcal{N}^{\text{RDMR}}$  the NFA  $\mathcal{N}$  is first reversed, then determinized, then minimized and then reversed again.

The átomaton has the following remarkable properties:

**Theorem 1 (Átomaton [34]).** *Let  $L$  be a regular language, let  $\mathcal{D}$  be its minimal DFA, and let  $\mathcal{A}$  be its átomaton. Then*

1.  $\mathcal{A}$  is isomorphic to  $\mathcal{D}^{\text{RDR}}$ .
2.  $\mathcal{A}^{\mathbb{R}}$  is isomorphic to the quotient DFA of  $L^R$ .
3.  $\mathcal{A}^{\mathbb{D}}$  is isomorphic to  $\mathcal{D}$ .
4. For any NFA  $\mathcal{N}$  accepting  $L$ ,  $\mathcal{N}^{\text{RDMR}}$  is isomorphic to  $\mathcal{A}$ .
5.  $\mathcal{A}$  is isomorphic to  $\mathcal{D}$  if and only if  $L$  is bideterministic.

A minimal DFA  $\mathcal{D}$  is *bideterministic* if its reverse is also a DFA. A language is *bideterministic* if its quotient DFA is bideterministic.

The quotient complexity of atoms of was computed in [33] using the átomaton. To find the complexity of atom  $A_i$ , the átomaton started in state  $A_i$  was converted to an equivalent DFA by the subset construction. A more direct and simpler method was used in [46] where the DFA accepting an atom of a given language is constructed directly from the DFA of the language.

It is clear that any language with  $n$  quotients has at most  $2^n$  atoms. It was proved in [33,46] that the following are upper bounds on the quotient complexities of atoms:

$$\kappa(A_S) \leq \begin{cases} 2^n - 1, & \text{if } S \in \{\emptyset, Q_n\}; \\ 1 + \sum_{x=1}^{|S|} \sum_{y=1}^{n-|S|} \binom{n}{x} \binom{n-x}{y}, & \text{if } \emptyset \subsetneq S \subsetneq Q_n. \end{cases}$$

It was shown in [33] that the language  $L_n$  of Fig. 1 has  $2^n$  atoms  $A_S$ , and each such atom meets the upper bound for the quotient complexity. On the other hand, the language  $L'_n = \Sigma^{n-2}$  has atoms:  $\Sigma^{n-2}, \Sigma^{n-3}, \dots, \Sigma, \varepsilon$ . Therefore  $L'_n$  has only  $n - 1$  atoms, and its most complex atom has complexity  $n$ . Hence atom complexity does distinguish well between  $L_n$  and  $L'_n$ . More will be said about atom complexity later.

The following property of the quotient complexity of atoms was proved by Diekert and Walter [38]. Let  $L_n$  be a language of quotient complexity  $n$ , and let  $f(n)$  be the maximal quotient complexity of its atoms. Then  $f(n+1)/f(n)$  approaches 3 as  $n$  approaches infinity.

## 6 Quotient Complexity of Operations

Many software systems have the capability of performing operations on regular languages represented by DFAs. For such systems it is necessary to know the maximal size of the result of the operation, to have some idea how long the computation will take and how much memory will be required. A lower bound on these time and space complexities is provided by the quotient/state complexity

of the result of the operation. For example, suppose we need to reverse a language  $L_n$ . We apply the reversal operation to a minimal DFA  $\mathcal{D}_n$  of  $L_n$  and then use the subset construction to determinize  $(\mathcal{D}_n)^\mathbb{R}$ . Since there are at most  $2^n$  reachable subsets, we know that  $2^n$  is an upper bound on the state complexity of reversal. Because we know that this bound can be reached,  $2^n$  is a lower bound on the time and space complexities of reversal.

From now on we denote a language of complexity  $n$  by  $L_n$ , and a DFA with  $n$  states, by  $\mathcal{D}_n$ . In general, the *complexity* of a regularity-preserving unary operation  $\circ$  on regular languages is the maximal value of  $\kappa(L_n^\circ)$  as a function of  $n$ , where  $L_n$  varies over all regular languages  $L_n$  with complexity  $n$ . To show that the bound is tight we need to exhibit a sequence  $(L_n, n \geq k) = (L_k, L_{k+1}, \dots)$ , called a *stream*, of languages that meet this bound. The stream does not necessarily start from 1, because the bound may not be reachable for small values of  $k$ . In the case of reversal, the stream  $(L_3, L_4, \dots)$  of Fig. 1 happens to meet the bound for  $n \geq 3$ .

In the case of star, Maslov [51] stated without proof that the tight upper bound for its complexity is  $2^{n-1} + 2^{n-2}$ . A proof was provided by Yu, Zhuang and Salomaa [63]. This bound is met by the DFA of Fig. 1 for  $n \geq 3$ .

Next consider the product  $L_m L_n$  of two languages  $L_m$  and  $L_n$ . Maslov stated without proof that the tight upper bound for product is  $(m-1)2^n + 2^{n-1}$ , and that this bound can be met. Yu, Zhuang and Salomaa [63] showed that there always exists a DFA with at most  $(m-1)2^n + 2^{n-1}$  states that accepts  $L_m L_n$ , and proved that the bound can be met. This bound is also met by  $L_m$  and  $L_n$  of Fig. 1 for  $m, n \geq 3$ .

In general, the *complexity* of a regularity-preserving binary operation  $\circ$  on regular languages of complexities  $m$  and  $n$ , respectively, is the maximal value of the result of the operation as a function of  $m$  and  $n$ , where the operands vary over all regular languages of complexities  $m$  and  $n$ , respectively. Thus we need two families  $(L'_{m,n} \mid m \geq h, n \geq k)$  and  $(L_{m,n} \mid m \geq h, n \geq k)$  of languages meeting this bound; the notation  $L'_{m,n}$  and  $L_{m,n}$  implies that  $L'_{m,n}$  and  $L_{m,n}$  depend on both  $m$  and  $n$ . Two such examples are known [42]: the union and intersection of finite languages require such witnesses. However, in all other cases studied in the literature, it is enough to use witness streams  $(L'_m, m \geq h)$  and  $(L_n, n \geq k)$ , where  $L'_m$  is independent of  $n$  and  $L_n$  is independent of  $m$ .

So far we have seen that the stream of Fig. 1 meets the upper bounds for syntactic complexity, quotients, atoms, reversal, star, and product. The situation is a little different for union (and other binary boolean operations). Since  $L_m \cup L_n$  can have at most  $mn$  quotients, we have an upper bound. Moreover, for  $m \neq n$ , we know [8] that the complexity of  $L_m \cup L_n$ , where these languages are defined in Fig. 1, does meet the bound  $mn$ . But because  $L_n \cup L_n = L_n$ , the complexity of union for the languages of Fig. 1 is  $n$  instead of  $n^2$ . So the same stream cannot be used for both arguments. However, it is possible to use a stream that “differs only slightly” from  $L_n$  of Fig. 1.

The notion “differs only slightly” is defined as follows [8,14,26]. Let  $\Sigma = \{a_1, \dots, a_k\}$  be an alphabet ordered as shown; if  $L \subseteq \Sigma^*$ , we denote it by



$L(a_1, \dots, a_k)$  to stress its dependence on  $\Sigma$ . A *dialect* of  $L$  is a language related to  $L$  and obtained by replacing or deleting letters of  $\Sigma$  in the words of  $L$ . More precisely, for an alphabet  $\Sigma'$  and a partial map  $\pi: \Sigma \mapsto \Sigma'$ , we obtain a dialect of  $L$  by replacing each letter  $a \in \Sigma$  by  $\pi(a)$  in every word of  $L$ , or deleting the word entirely if  $\pi(a)$  is undefined. We write  $L(\pi(a_1), \dots, \pi(a_k))$  to denote the dialect of  $L(a_1, \dots, a_k)$  given by  $\pi$ , and we denote undefined values of  $\pi$  by “—”. For example, if  $L(a, b, c) = \{a, ab, ac\}$  then its dialect  $L(b, -, d)$  is the language  $\{b, bd\}$ . Undefined values for letters at the end of the alphabet are omitted; thus, for example, if  $\Sigma = \{a, b, c, d, e\}$ ,  $\pi(a) = b$ ,  $\pi(b) = a$ ,  $\pi(c) = c$  and  $\pi(d) = \pi(e) = -$ , we write  $L(b, a, c)$  for  $L(b, a, c, -, -)$ .

In general, for any binary boolean operation  $\circ$  on languages  $L_m$  and  $L_n$  with quotient DFAs  $\mathcal{D}_m$  and  $\mathcal{D}_n$ , to find  $L_m \circ L_n$  we use the direct product of  $\mathcal{D}_m$  and  $\mathcal{D}_n$  and assign final states in the direct product according to the operation  $\circ$ . This gives an upper bound of  $mn$  for all the operations. If we know that the bound  $mn$  is met by  $L_m \cup L_n$ , we also know that the intersection  $\overline{L_m} \cap \overline{L_n}$  meets that bound, because  $\kappa(\overline{L}) = \kappa(L)$  for all  $L$ ; similarly, the difference  $L_m \setminus \overline{L_n}$  meets that bound. It is also known that there are witnesses  $L_m$  and  $L_n$  such that the symmetric difference  $L_m \oplus L_n$  meets the bound  $mn$ . A binary boolean function  $\circ$  is *proper* if it depends on both of its arguments. There are six more proper boolean functions:  $\overline{K} \cup \overline{L} = \overline{K \cap L}$ ,  $\overline{K} \cap \overline{L} = \overline{K \cup L}$ ,  $\overline{K} \cup L = \overline{K \setminus L}$ ,  $\overline{K} \cap L = \overline{L \setminus K}$ ,  $K \cup \overline{L} = \overline{L \setminus K}$ , and  $\overline{K \oplus L}$ . Thus witnesses for these six functions can be found using the witnesses for union and symmetric difference and their complements.

Our discussion so far, as well as all the literature prior to 2016, used witnesses *restricted* to the same alphabet. However it is also useful to perform binary operations on languages over different alphabets, for example:  $\{ab\}\{ac\}$  or  $\{a, b\}^*b \cup \{a, c\}^*c$ . The *unrestricted* complexity of binary operations was first studied in [10]. In the case of union and symmetric difference of  $L'_m \subseteq (\Sigma')^*$  and  $L_n \subseteq \Sigma^*$ , the result is a language over the alphabet  $\Sigma' \cup \Sigma$ . To compute the complexity of  $L'_m \cup L_n$ , if  $L'_m$  does not have an empty quotient, we add an empty state to  $\mathcal{D}'_m$  and send all transitions under letters from  $\Sigma \setminus \Sigma'$  to that state. Similarly, we add an empty state if needed to  $\mathcal{D}_n$  and send all transitions under letters from  $\Sigma' \setminus \Sigma$  to that state. Thus we have now two languages over the alphabet  $\Sigma' \cup \Sigma$ , and we proceed as in the restricted case over the larger alphabet. It turns out that the complexity of union and symmetric difference is  $(m+1)(n+1)$  [10].

For difference and intersection,  $(m+1)(n+1)$  is still an upper bound on their complexity. However, the alphabet of  $L'_m \setminus L_n$  is  $\Sigma'$  and the complexity turns out to be  $mn + m$  for the difference operation. Similarly, the alphabet of  $L'_m \cap L_n$  is  $\Sigma' \cap \Sigma$ , and the complexity of intersection is  $mn$ , as in the restricted case. The complexity of any other binary boolean operation can be determined from the complexities of union, intersection, difference and symmetric difference; however, the complexity of  $L'_m \circ L_n$  may differ by 1 from the complexity of  $\overline{L'_m \circ L_n}$ . For more details see [24].

## 7 Complexity Measures

We have introduced the following measures of complexity for regular languages  $L_m$  and  $L_n$  [8,10]:

1. The size of the syntactic semigroup of  $L_n$ .
2. The complexity of the quotients of  $L_n$ .
3. The number of atoms of  $L_n$ .
4. The complexity of the atoms of  $L_n$ .
5. The complexity of the reverse  $L_n^R$  of  $L_n$ .
6. The complexity of  $L_n^*$ , the star of  $L_n$ .
7. The restricted and unrestricted complexities of the product  $L_m L_n$ .
8. The restricted and unrestricted complexities of boolean operations  $L'_m \circ L_n$ .

These measures are not all independent: the relations described below are known.

**Theorem 2 (Semigroup and Reversal [57]).** *Let  $\mathcal{D}$  be a minimal DFA with  $n$  states accepting a language  $L$ . If the transition semigroup of  $\mathcal{D}$  has  $n^n$  elements, then the complexity of  $L^R$  is  $2^n$ .*

**Theorem 3 (Number of Atoms and Reversal [34]).** *The number of atoms of a regular language  $L$  is equal to the complexity of  $L^R$ .*

Before discussing the next relationships we need to introduce certain concepts from group theory. If  $G$  is a permutation group,  $G$  is *transitive* on a set  $X$  if for all  $x, y \in X$ , there exists  $g \in G$  such that  $xg = y$ . Also,  $G$  is *k-set-transitive* if it is transitive on the set of  $k$ -subsets of  $Q_n$ , that is, if for all  $X, Y \subseteq Q_n$  such that  $|X| = |Y| = k$ , there exists  $g \in G$  such that  $Xg = Y$ . If  $G$  has degree  $n$  and is  $k$ -set-transitive for  $0 \leq k \leq n$ , then  $G$  is *set-transitive*.

Set transitive groups have been characterized as follows:

**Theorem 4 (Set Transitive Groups [2]).** *A set-transitive permutation group of degree  $n$  is  $S_n$  or  $A_n$  or a conjugate of one of the following permutation groups:*

1. For  $n = 5$ , the affine general linear group  $\text{AGL}(1, 5)$ .
2. For  $n = 6$ , the projective general linear group  $\text{PGL}(2, 5)$ .
3. For  $n = 9$ , the projective special linear group  $\text{PSL}(2, 8)$ .
4. For  $n = 9$ , the projective semilinear group  $\text{PTL}(2, 8)$ .

We say  $L$  is maximally atomic if it has the maximal number of atoms, and each of those atoms has the maximal possible complexity. The *rank* of a transformation  $t$  is the cardinality of  $Q_n t$ . The next result characterizes maximally atomic languages.

**Theorem 5 (Maximally Atomic Languages [11]).** *Let  $L$  be a regular language over  $\Sigma$  with complexity  $n \geq 3$ , and let  $T$  be the transition semigroup of the minimal DFA of  $L$ . Then  $L$  is maximally atomic if and only if the subgroup of permutations in  $T$  is set-transitive, and  $T$  contains a transformation of rank  $n - 1$ .*

Define the following classes of languages:

- **FTS** - languages whose minimal DFAs have the full transformation semigroup of  $n^n$  elements.
- **STS** - languages whose minimal DFAs have transition semigroups with a set-transitive subgroup of permutations and a transformation of rank  $n - 1$ .
- **MAL** - maximally atomic languages.
- **MNA** - languages with the maximal number of atoms.
- **MCR** - languages with a maximally complex reverse.

The known relations among the various complexity measures are thus as follows:

$$\mathbf{FTS} \subset \mathbf{STS} = \mathbf{MAL} \subset \mathbf{MNA} = \mathbf{MCR}$$

## 8 Most Complex Regular Language Streams

We now exhibit a regular language stream that, together with some dialects, meets the upper bounds for all complexity measures we have discussed so far [8,24]. In this sense this is a *most complex regular language stream* or a *universal witness stream*. This stream differs from the stream of Fig. 1 only by the identity input  $d$ .

**Definition 1.** For  $n \geq 3$ , let  $\mathcal{D}_n = \mathcal{D}_n(a, b, c, d) = (Q_n, \Sigma, \delta_n, 0, \{n - 1\})$ , where  $\Sigma = \{a, b, c, d\}$ , and  $\delta_n$  is defined by the transformations  $a: (0, \dots, n - 1)$ ,  $b: (0, 1)$ ,  $c: (n - 1 \rightarrow 0)$ , and  $d: \mathbb{1}$ . Let  $L_n = L_n(a, b, c, d)$  be the language accepted by  $\mathcal{D}_n$ .

**Theorem 6 ( Most Complex Regular Languages).** For each  $n \geq 3$ , the DFA of Definition 1 is minimal and its language  $L_n(a, b, c, d)$  has complexity  $n$ . The stream  $(L_n(a, b, c, d) \mid n \geq 3)$  with some dialect streams is most complex in the class of regular languages. In particular, it meets all the complexity bounds below, which are maximal for regular languages. In several cases the bounds can be met with a reduced alphabet.

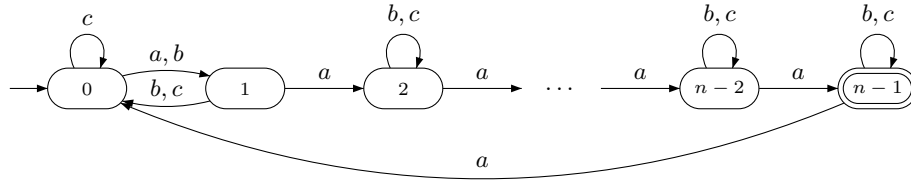
1. The syntactic semigroup of  $L_n(a, b, c)$  has cardinality  $n^n$ , and at least three letters are required to meet this bound.
2. Each quotient of  $L_n(a)$  has complexity  $n$ .
3. The reverse of  $L_n(a, b, c)$  has complexity  $2^n$ , and  $L_n(a, b, c)$  has  $2^n$  atoms.
4. For each atom  $A_S$  of  $L_n(a, b, c)$ , the complexity  $\kappa(A_S)$  satisfies:  $\kappa(A_S) = 2^n - 1$ , if  $S \in \{\emptyset, Q_n\}$ ;  $\kappa(A_S) = 1 + \sum_{x=1}^{|S|} \sum_{y=1}^{n-|S|} \binom{n}{x} \binom{n-x}{y}$ , if  $\emptyset \subsetneq S \subsetneq Q_n$ .
5. The star of  $L_n(a, b)$  has complexity  $2^{n-1} + 2^{n-2}$ .
6. Product
  - (a) Restricted:  $\kappa(L_m(a, b, c)L_n(a, b, c)) = m2^n - 2^{n-1}$ .
  - (b) Unrestricted:  $\kappa(L_m(a, b, -, c)L_n(b, a, -, d)) = m2^n + 2^{n-1}$ .
7. Boolean operations
  - (a) Restricted: For any proper binary boolean operation  $\circ$ ,  $\kappa(L_m(a, b) \circ L_n(b, a)) = mn$ .

- (b) *Unrestricted:*  $\kappa(L_m(a, b, -, c) \circ L_n(b, a, -, d)) = (m+1)(n+1)$  if  $\circ \in \{\cup, \oplus\}$ ,  $\kappa(L_m(a, b, -, c) \setminus L_n(b, a)) = mn+m$ , and  $\kappa(L_m(a, b) \cap L_n(b, a)) = mn$ .

At least four letters are necessary for unrestricted operations [10].

In the stream above we have used a “master language”  $L_n$  of Definition 1 with four letters, and dialects that use the same alphabet as the master language. The stream below uses only three letters in the master language of Definition 2, but then adds an extra letter  $d$  in a dialect.

**Definition 2.** For  $n \geq 3$ , let  $\mathcal{D}_n = \mathcal{D}_n(a, b, c) = (Q_n, \Sigma, \delta_n, 0, \{n-1\})$ , where  $\Sigma = \{a, b, c\}$ , and  $\delta_n$  is defined by the transformations  $a: (0, \dots, n-1)$ ,  $b: (0, 1)$ , and  $c: (1 \rightarrow 0)$ . Let  $L_n = L_n(a, b, c)$  be the language accepted by  $\mathcal{D}_n$ . The structure of  $\mathcal{D}_n(a, b, c)$  is shown in Fig. 2.



**Fig. 2.** Minimal DFA of a most complex regular language.

The properties of  $L_n$  are the same as those in Theorem 6 except for the following:

- The bound for the restricted product is met by  $L_m(a, b)L_n(a, -, b)$ .
- The bound for the unrestricted product is met by  $L_m(a, b)L_n(a, c, b)$ .
- The bound for the unrestricted union and symmetric difference is met by  $L_m(a, b, c)L_n(b, a, d)$ .
- The bound for the unrestricted difference is met by  $L_m(a, b, c)L_n(b, a)$ .

The most complex streams introduced in this section will be used in several subclasses of regular languages.

## 9 Most Complex Languages in Subclasses

Many interesting proper subclasses of the class of regular languages can be defined using the notion of convexity. Convex languages were introduced in 1973 by Thierrin [61] and revisited in 2009 by Ang and Brzozowski [1].

Convexity can be defined with respect to any binary relation on  $\Sigma^*$ . Let  $\trianglelefteq$  be such a binary relation; if  $u \trianglelefteq v$  and  $u \neq v$ , we write  $u \triangleleft v$ . Let  $\trianglerighteq$  be the

converse binary relation, that is, let  $u \supseteq v$  if and only if  $v \preceq u$ . A language  $L$  is  $\preceq$ -convex if  $u \preceq v$ ,  $u \preceq w$ , and  $v \preceq w$  with  $u, w \in L$  imply  $v \in L$ . It is  $\preceq$ -free if  $v \preceq w$  and  $w \in L$  imply  $v \notin L$ . It is  $\preceq$ -closed if  $v \preceq w$  and  $w \in L$  imply  $v \in L$ . It is  $\supseteq$ -closed if  $v \supseteq w$  and  $w \in L$  imply  $v \in L$ . Languages that are  $\supseteq$ -closed are also called  $\preceq$ -converse-closed. One verifies that a language is  $\preceq$ -closed if and only if its complement is  $\supseteq$ -closed.

If  $w = xyz$ , where  $x, y, z \in \Sigma^*$ , then  $x$  is a *prefix* of  $w$ ,  $y$  is a *factor* of  $w$ , and  $z$  is a *suffix* of  $w$ . Note that a prefix or a suffix is also a factor. If  $w = w_0 a_1 w_1 \cdots a_n w_n$ , where  $a_1, \dots, a_n \in \Sigma$ , and  $w_0, \dots, w_n \in \Sigma^*$ , then  $v = a_1 \cdots a_n$  is a *subword* of  $w$ ; note that every factor of  $w$  is a subword<sup>3</sup> of  $w$ .

The *shuffle*  $u \sqcup v$  of words  $u, v \in \Sigma^*$  is defined as follows:

$$u \sqcup v = \{u_1 v_1 \cdots u_k v_k \mid u = u_1 \cdots u_k, v = v_1 \cdots v_k, u_1, \dots, u_k, v_1, \dots, v_k \in \Sigma^*\}.$$

The shuffle of two languages  $K$  and  $L$  over  $\Sigma$  is defined by

$$K \sqcup L = \bigcup_{u \in K, v \in L} u \sqcup v.$$

Note that the shuffle operation is commutative on both words and languages.

Here we consider only four binary relations for defining convexity: “is a prefix of”, “is a suffix of”, “is a factor of”, and “is a subword of”. Each of these four relations is a partial order on  $\Sigma^*$  and leads to four classes of languages; we illustrate this using the prefix relation:

- A language  $L$  that is *prefix-converse-closed* is a *right ideal*, that is, it satisfies the equation  $L = L\Sigma^*$ .
- A language  $L$  that is *prefix-closed* is the complement of a right ideal.
- A language that is *prefix-free* and not  $\{\varepsilon\}$  is a prefix-code [4].
- A language is *proper prefix-convex* if it not a right ideal and is neither closed nor free.

Similarly, we define *suffix-converse-closed* languages which are *left ideals* (satisfy  $L = \Sigma^* L$ ), *suffix-closed*, *suffix-free* (*suffix codes* [4]), and *proper suffix-convex* languages, *two-sided ideals* (that satisfy  $L = \Sigma^* L \Sigma^*$ ), *factor-closed*, *factor-free* (*infix codes* [59]), and *proper factor-convex* languages, and also *subword-converse-closed* languages which are *all-sided ideals* (that satisfy  $L = L \sqcup \Sigma^*$ ), *subword-closed*, *subword-free* (*hypercodes* [59]), and *proper subword-convex* languages.

Decision problems for convex languages were studied in [22]. We can decide in  $O(n^3)$  time if a given regular language  $L$  over a fixed alphabet  $\Sigma$  accepted by a DFA with  $n$  states is prefix-, suffix-, factor-, and subword-convex. We can decide in  $O(n^2)$  time if  $L$  is prefix-free, left ideal, suffix-closed, suffix-free, two-sided ideal, factor-closed, factor-free, all-sided ideal, subword-closed, subword-free. We can decide in  $O(n)$  time if  $L$  is a right ideal or a prefix-closed language.

We now consider the complexity properties of some convex languages.

<sup>3</sup> The word “subword” is often used to mean “factor”; here by a “subword” we mean a subsequence.

### 9.1 Prefix-Convex Languages

**RIGHT IDEALS** The complexity of right ideals was studied as follows: complexities of common operations using various witnesses [15], semigroup size [35], complexities of atoms [12], most complex right ideals with restricted operations [14], most complex right ideals with restricted and unrestricted operations and four-letter witnesses [26], most complex right ideals with restricted and unrestricted operations and five-letter witnesses [24]. Here we use the witnesses from [26].

**Definition 3.** For  $n \geq 4$ , let  $\mathcal{D}_n = \mathcal{D}_n(a, b, c, d) = (Q_n, \Sigma, \delta_n, 0, \{n-1\})$ , where  $\Sigma = \{a, b, c, d\}$  and  $\delta_n$  is defined by  $a: (0, \dots, n-2)$ ,  $b: (0, 1)$ ,  $c: (1 \rightarrow 0)$ , and  $d: \binom{n-2}{0} q \rightarrow q+1$ . This DFA uses the structure of Fig. 2 for the states in  $Q_{n-1} = \{0, \dots, n-2\}$  and letters in  $\{a, b, c\}$ . Let  $L_n = L_n(a, b, c, d)$  be the language of  $\mathcal{D}_n$ .

**Theorem 7 (Most Complex Right Ideals).** For each  $n \geq 4$ , the DFA of Definition 3 is minimal and  $L_n(a, b, c, d)$  is a right ideal of complexity  $n$ . The stream  $(L_n(a, b, c, d) \mid n \geq 4)$  with some dialect streams is most complex in the class of right ideals. It meets the following bounds: 1. Semigroup size:  $n^{n-1}$ . 2. Quotient complexities:  $n$ , except  $\kappa(\Sigma^*) = 1$ . 3. Reversal:  $2^{n-1}$ . 4. Atom complexities:  $\kappa(A_S) = 2^{n-1}$ , if  $S = Q_n$ ;  $\kappa(A_S) = 1 + \sum_{x=1}^{|S|} \sum_{y=1}^{n-|S|} \binom{n-1}{x-1} \binom{n-x}{y}$ , if  $\emptyset \subsetneq S \subsetneq Q_n$ . 5. Star:  $n+1$ . 6. (a) Restricted product:  $m + 2^{n-2}$ ; (b) Unrestricted product:  $m + 2^{n-1} + 2^{n-2} + 1$ . 7. (a) Restricted boolean operations:  $mn$  if  $\circ \in \{\cap, \oplus\}$ ,  $mn - (m-1)$  if  $\circ = \setminus$ , and  $mn - (m+n-2)$  if  $\circ = \cup$ . (b) Unrestricted boolean operations: same as regular languages. At least four letters are required to meet all these bounds [32].

**PREFIX-CLOSED LANGUAGES** The complexities of common operations on prefix-closed languages using various witnesses were studied in [17,43]. Most complex prefix-closed languages were examined in [26]. As every prefix-closed language has an empty quotient, the restricted and unrestricted complexities are the same.

**Definition 4.** For  $n \geq 4$ , let  $\mathcal{D}_n = \mathcal{D}_n(a, b, c, d) = (Q_n, \Sigma, \delta_n, 0, Q_n \setminus \{n-1\})$ , where  $\Sigma = \{a, b, c, d\}$ , and  $\delta_n$  is defined by  $a: (0, \dots, n-2)$ ,  $b: (0, 1)$ ,  $c: (1 \rightarrow 0)$ , and  $d: \binom{0}{n-2} q \rightarrow q-1 \pmod{n}$ . Let  $L_n = L_n(a, b, c, d)$  be the language of  $\mathcal{D}_n$ .

**Theorem 8 (Most Complex Prefix-Closed Languages).** For  $n \geq 4$ , the DFA of Definition 4 is minimal and  $L_n(\Sigma_n)$  is a prefix-closed language of complexity  $n$ . The stream  $(L_m(a, b, c, d) \mid m \geq 4)$  with some dialect streams is most complex in the class of prefix-closed languages, and meets the following bounds: 1. Semigroup size:  $n^{n-1}$ . 2. Quotient complexities:  $n$ , except  $\kappa(\emptyset) = 1$ . 3. Reversal:  $2^{n-1}$ . 4. Atom complexities:  $\kappa(A_S) = 2^{n-1}$ , if  $S = \emptyset$ ;  $\kappa(A_S) = 1 + \sum_{x=1}^{n-|S|} \sum_{y=1}^{|S|} \binom{n-1}{x-1} \binom{n-x}{y}$ , if  $\emptyset \subsetneq S \subsetneq Q_n$ . 5. Star  $2^{n-2} + 1$ . 6. Product:  $(m+1)2^{n-2}$ . 7. Boolean operations:  $mn$  if  $\circ \in \{\cup, \oplus\}$ ,  $mn - (n-1)$  if  $\circ = \setminus$ , and  $mn - (m+n-2)$  if  $\circ = \cap$ . At least four letters are required to meet all these bounds [26].

**PREFIX-FREE LANGUAGES** The complexities of operations on prefix-free languages with various witnesses were studied in [43,47,50]. The syntactic complexity bound of  $n^{n-2}$  was established in [20]. Most complex prefix-free languages were considered in [26]. As every prefix-free language has an empty quotient, the restricted and unrestricted complexities are the same for binary operations.

**Definition 5.** For  $n \geq 4$ , let  $\Sigma_n = \{a, b, c, d, e_0, \dots, e_{n-3}\}$  and let DFA  $\mathcal{D}_n(\Sigma_n)$  be  $\mathcal{D}_n(\Sigma_n) = (Q_n, \Sigma_n, \delta_n, 0, \{n-2\})$ , where  $\delta_n$  is defined by  $a: (n-2 \rightarrow n-1)(0, \dots, n-3)$ ,  $b: (n-2 \rightarrow n-1)(0, 1)$ ,  $c: (n-2 \rightarrow n-1)(1 \rightarrow 0)$ ,  $d: (0 \rightarrow n-2)(Q_n \setminus \{0\} \rightarrow n-1)$ ,  $e_q: (n-2 \rightarrow n-1)(q \rightarrow n-2)$  for  $q = 0, \dots, n-3$ . The transformations induced by  $a$  and  $b$  coincide when  $n = 4$ . This DFA uses the structure of the DFA of Fig. 2 for the states in  $Q_{n-2} = \{0, \dots, n-3\}$  and letters in  $\{a, b, c\}$ . Let  $L_n(\Sigma_n)$  be the language of  $\mathcal{D}_n(\Sigma_n)$ .

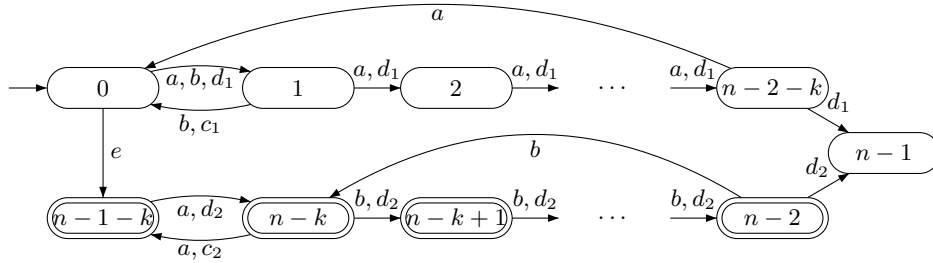
**Theorem 9 (Most Complex Prefix-Free Languages).** For  $n \geq 4$ , the DFA of Definition 5 is minimal and  $L_n(\Sigma_n)$  is a prefix-free language of complexity  $n$ . The stream  $(L_n(a, b, c, d, e_0, \dots, e_{n-3}) \mid n \geq 4)$  with some dialect streams is a most complex prefix-free language. At least  $n+2$  inputs are required to meet all the bounds below [26]: 1. Semigroup size:  $n^{n-2}$ . 2. Quotient complexities:  $n$ , except  $\kappa(\varepsilon) = 2$ ,  $\kappa(\emptyset) = 1$ . 3. Reversal:  $2^{n-2} + 1$ . 4. Atom complexities:  $\kappa(A_S) = 2$ , if  $S = \{n-2\}$ ;  $\kappa(A_S) = 2^{n-1}$ , if  $S = \emptyset$ ;  $\kappa(A_S) = 2^{n-2} + 1$ , if  $S = Q_{n-2}$ ;  $\kappa(A_S) = 2 + \sum_{x=1}^{|S|} \sum_{y=1}^{n-2-|S|} \binom{n-2}{x} \binom{n-2-x}{y}$ , if  $\emptyset \subsetneq S \subsetneq Q_{n-2}$ . 5. Star:  $n$ . 6. Product:  $m + n - 2$ . 7. Boolean operations:  $mn - 2$  if  $\circ \in \{\cup, \oplus\}$ ,  $mn - (m + 2n - 4)$  if  $\circ = \setminus$ , and  $mn - 2(m + n - 3)$  if  $\circ = \cap$ .

**PROPER PREFIX- CONVEX LANGUAGES** Proper prefix-convex languages were studied in [23]. In contrast to the three special cases, they represent the full nature of prefix-convexity.

**Definition 6.** For  $n \geq 3$ ,  $1 \leq k \leq n-2$ , let  $\mathcal{D}_{n,k}(\Sigma) = (Q_n, \Sigma, \delta_{n,k}, 0, F_{n,k})$  where  $\Sigma = \{a, b, c_1, c_2, d_1, d_2, e\}$ ,  $F_{n,k} = \{n-1-k, \dots, n-2\}$ , and  $\delta_{n,k}$  is given by the transformations below.

Also, let  $E_{n,k} = \{0, \dots, n-2-k\}$ ; it is useful to partition  $Q_n$  into  $E_{n,k}$ ,  $F_{n,k}$ , and  $\{n-1\}$ . Letters  $a$  and  $b$  have complementary behaviours on  $E_{n,k}$  and  $F_{n,k}$ , depending on the parities of  $n$  and  $k$ . Letters  $c_1$  and  $d_1$  act on  $E_{n,k}$  exactly in the same way as  $c_2$ , and  $d_2$  act on  $F_{n,k}$ . In addition,  $d_1$  and  $d_2$  send states  $n-2-k$  and  $n-2$ , respectively, to state  $n-1$ , and letter  $e$  connects the two parts of the DFA. The structure of  $\mathcal{D}_n(\Sigma)$  is shown in Figs. 3 and 4 for certain parities of  $n-1-k$  and  $k$ . Let  $L_{n,k}(\Sigma)$  be the language recognized by  $\mathcal{D}_{n,k}(\Sigma)$ .

$$\begin{aligned}
a: & \begin{cases} (1, \dots, n-2-k)(n-1-k, n-k), & \text{if } n-1-k \text{ is even and } k \geq 2; \\ (0, \dots, n-2-k)(n-1-k, n-k), & \text{if } n-1-k \text{ is odd and } k \geq 2; \\ (1, \dots, n-2-k), & \text{if } n-1-k \text{ is even and } k = 1; \\ (0, \dots, n-2-k), & \text{if } n-1-k \text{ is odd and } k = 1. \end{cases} \\
b: & \begin{cases} (n-k, \dots, n-2)(0, 1), & \text{if } k \text{ is even and } n-1-k \geq 2; \\ (n-1-k, \dots, n-2)(0, 1), & \text{if } k \text{ is odd and } n-1-k \geq 2; \\ (n-k, \dots, n-2), & \text{if } k \text{ is even and } n-1-k = 1; \\ (n-1-k, \dots, n-2), & \text{if } k \text{ is odd and } n-1-k = 1. \end{cases} \\
c_1: & \begin{cases} (1 \rightarrow 0), & \text{if } n-1-k \geq 2; \\ \mathbb{1}, & \text{if } n-1-k = 1. \end{cases} \\
c_2: & \begin{cases} (n-k \rightarrow n-1-k), & \text{if } k \geq 2; \\ \mathbb{1}, & \text{if } k = 1. \end{cases} \\
d_1: & (n-2-k \rightarrow n-1) \binom{n-3-k}{0} q \rightarrow q+1. \\
d_2: & \binom{n-2}{n-1-k} q \rightarrow q+1. \\
e: & (0 \rightarrow n-1-k).
\end{aligned}$$

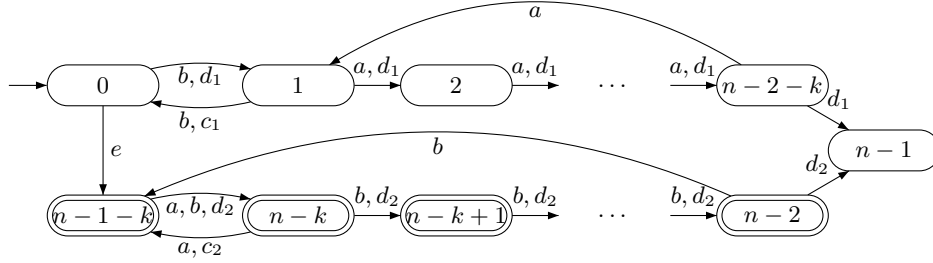


**Fig. 3.** DFA  $\mathcal{D}_{n,k}(a, b, c_1, c_2, d_1, d_2, e)$  of Definition 6 when  $n-1-k$  is odd,  $k$  is even, and both are at least 2; missing transitions are self-loops.

**Theorem 10 (Proper Prefix-Convex Languages).** *For  $n \geq 3$  and  $1 \leq k \leq n-2$ , the DFA  $\mathcal{D}_{n,k}(\Sigma)$  of Definition 6 is minimal and  $L_{n,k}(\Sigma)$  is a  $k$ -proper language of complexity  $n$ . The bounds below are maximal for  $k$ -proper prefix-convex languages. At least seven letters are required to meet these bounds.*

1. *The syntactic semigroup of  $L_{n,k}(\Sigma)$  has cardinality  $n^{n-1-k}(k+1)^k$ ; the maximal value  $n(n-1)^{n-2}$  is reached only when  $k = n-2$ .*





**Fig. 4.** DFA  $\mathcal{D}_{n,k}(a, b, c_1, c_2, d_1, d_2, e)$  of Definition 6 when  $n - 1 - k$  is even,  $k$  is odd, and both are at least 2; missing transitions are self-loops.

2. The non-empty, non-final quotients of  $L_{n,k}(a, b, -, -, -, d_2, e)$  have complexity  $n$ , the final quotients have complexity  $k + 1$ , and  $\emptyset$  has complexity 1.
3. The reverse of  $L_{n,k}(a, b, -, -, -, d_2, e)$  has complexity  $2^{n-1}$ ; moreover, the language  $L_{n,k}(a, b, -, -, -, d_2, e)$  has  $2^{n-1}$  atoms for all  $k$ .
4. For each atom  $A_S$  of  $L_{n,k}(\Sigma)$ , write  $S = X_1 \cup X_2$ , where  $X_1 \subseteq E_{n,k}$  and  $X_2 \subseteq F_{n,k}$ . Let  $\bar{X}_1 = E_{n,k} \setminus X_1$  and  $\bar{X}_2 = F_{n,k} \setminus X_2$ . If  $X_2 \neq \emptyset$ , then  $\kappa(A_S) = 1 + \sum_{x_1=0}^{|X_1|} \sum_{x_2=1}^{|X_2|-x_1} \sum_{y_1=0}^{|\bar{X}_1|} \sum_{y_2=0}^{|\bar{X}_2|-y_1} \binom{n-1-k}{x_1} \binom{k}{x_2} \binom{n-1-k-x_1}{y_1} \binom{k-x_2}{y_2}$ . If  $X_1 \neq \emptyset$  and  $X_2 = \emptyset$ , then  $\kappa(A_S) = 1 + \sum_{x_1=0}^{|X_1|} \sum_{x_2=0}^{|X_1|-x_1} \sum_{y_1=0}^{|\bar{X}_1|} \sum_{y_2=0}^k \binom{n-1-k}{x_1} \binom{k}{x_2} \binom{n-1-k-x_1}{y_1} \binom{k-x_2}{y_2} - 2^k \sum_{y=0}^{|\bar{X}_1|} \binom{n-1-k}{y}$ . Otherwise,  $S = \emptyset$  and  $\kappa(A_S) = 2^{n-1}$ .
5. The star of  $L_{n,k}(a, b, -, -, d_1, d_2, e)$  has complexity  $2^{n-2} + 2^{n-2-k} + 1$ . The maximal value  $2^{n-2} + 2^{n-3} + 1$  is reached only when  $k = 1$ .
6.  $L_{m,j}(a, b, c_1, -, d_1, d_2, e) L_{n,k}(a, d_2, c_1, -, d_1, b, e)$  has complexity  $m - 1 - j + j2^{n-2} + 2^{n-1}$ . The maximal value  $m2^{n-2} + 1$  is reached only when  $j = m - 2$ .
7. For  $m, n \geq 3$ ,  $1 \leq j \leq m - 2$ , and  $1 \leq k \leq n - 2$ , define the languages  $L_{m,j} = L_{m,j}(a, b, c_1, -, d_1, d_2, e)$  and  $L_{n,k} = L_{n,k}(a, b, e, -, d_2, d_1, c_1)$ . For any proper binary boolean function  $\circ$ , the complexity of  $L_{m,j} \circ L_{n,k}$  is maximal. Thus
  - (a)  $L_{m,j} \cup L_{n,k}$  and  $L_{m,j} \oplus L_{n,k}$  have complexity  $mn$ .
  - (b)  $L_{m,j} \setminus L_{n,k}$  has complexity  $mn - (n - 1)$ .
  - (c)  $L_{m,j} \cap L_{n,k}$  has complexity  $mn - (m + n - 2)$ .

## 9.2 Suffix-Convex Languages

**LEFT IDEALS** The complexity of left ideals was studied as follows: complexities of common operations using various witnesses [15], semigroup size lower bound [35], semigroup size upper bound [28], complexities of atoms [12], most complex left ideals with restricted operations [14], most complex left ideals with restricted and unrestricted operations [24].

**Definition 7.** For  $n \geq 4$ , let  $\mathcal{D}_n = \mathcal{D}_n(a, b, c, d, e) = (Q_n, \Sigma, \delta_n, 0, \{n - 1\})$ , where  $\Sigma = \{a, b, c, d, e\}$ , and  $\delta_n$  is defined by transformations  $a: (1, \dots, n - 1)$ ,

$b: (1, 2)$ ,  $c: (n-1 \rightarrow 1)$ ,  $d: (n-1 \rightarrow 0)$ , and  $e: (Q_n \rightarrow 1)$ . Denote by  $L_n = L_n(a, b, c, d, e)$  the language accepted by  $\mathcal{D}_n$ .

**Theorem 11 (Most Complex Left Ideals).** *For each  $n \geq 4$ , the DFA of Definition 7 is minimal, and its language is a left ideal of complexity  $n$ . The stream  $(L_n(a, b, c, d, e) \mid n \geq 4)$  with some dialect streams is most complex in the class of regular left ideals as follows: 1. Semigroup size:  $n^{n-1} + n - 1$ . 2. Quotient complexities:  $n$ . 3. Reversal:  $2^{n-1} + 1$ . 4. Atom complexities:  $\kappa(A_S) = n$ , if  $S = Q_n$ ;  $\kappa(A_S) = 2^{n-1}$ , if  $S = \emptyset$ ;  $\kappa(A_S) = 1 + \sum_{x=1}^{|S|} \sum_{y=1}^{n-|S|} \binom{n-1}{x} \binom{n-x-1}{y-1}$ , otherwise. 5. Star:  $n + 1$ . 6. (a) Restricted product:  $m + n + 1$ ; (b) unrestricted product:  $mn + m + n$ . 7. Restricted and unrestricted boolean operations: same as regular languages. At least five letters are required to meet all these bounds [32].*

**SUFFIX-CLOSED LANGUAGES** The complexities of common operations using various witnesses were studied in [17], and most complex suffix-closed languages in [25].

**Definition 8.** *For  $n \geq 4$ , let  $\mathcal{D}_n = \mathcal{D}_n(a, b, c, d, e) = (Q_n, \Sigma, \delta_n, 0, \{0\})$ , where  $\Sigma = \{a, b, c, d, e\}$ , and  $\delta_n$  is defined by transformations  $a: (1, \dots, n-1)$ ,  $b: (1, 2)$ ,  $c: (n-1 \rightarrow 1)$ ,  $d: (n-1 \rightarrow 0)$ ,  $e: (Q_n \rightarrow 1)$ . Let  $L_n = L_n(a, b, c, d, e)$  be the language of  $\mathcal{D}_n$ .*

**Theorem 12 (Most Complex Suffix-Closed Languages).** *For each  $n \geq 4$ , the DFA of Definition 8 is minimal and its language  $L_n(a, b, c, d, e)$  is suffix-closed and has complexity  $n$ . The stream  $(L_n(a, b, c, d, e) \mid n \geq 4)$  with some dialect streams is most complex in the class of suffix-closed languages. 1. Semigroup size:  $n^{n-1} + n - 1$ . 2. Quotients:  $n$ . 3. Reversal:  $2^{n-1} + 1$ . 4. Atom complexities:  $\kappa(A_S) = n$ , if  $S = \emptyset$ ;  $\kappa(A_S) = 2^{n-1}$ , if  $S = Q_n$ ;  $\kappa(A_S) = 1 + \sum_{x=1}^{|S|} \sum_{y=1}^{n-|S|} \binom{n-1}{y} \binom{n-y-1}{x-1}$ , otherwise. 5. Star:  $n$ . 6. (a) Restricted product:  $mn - n + 1$ ; (b) unrestricted product:  $mn + m + 1$ . 7. Restricted and unrestricted boolean operations: same as regular languages.*

**SUFFIX-FREE LANGUAGES** The complexities of common operations using various witnesses were studied in [25, 30, 36, 44, 48], semigroup size lower bound in [20], and upper bound in [31]. Suffix-free languages were the first example found of a class in which a most complex stream does not exist [30]. However, two streams cover all the complexity measures [30]. Since every suffix-free language has an empty quotient, the restricted and unrestricted cases for binary operations coincide.

**Definition 9.** *For  $n \geq 4$ , define the DFA  $\mathcal{D}_n(a, b, c, d, e) = (Q_n, \Sigma, \delta, 0, F)$ , where  $Q_n = \{0, \dots, n-1\}$ ,  $\Sigma = \{a, b, c, d, e\}$ ,  $\delta$  is given by  $a: (0 \rightarrow n-1)(1, \dots, n-2)$ ,  $b: (0 \rightarrow n-1)(1, 2)$ ,  $c: (0 \rightarrow n-1)(n-2 \rightarrow 1)$ ,  $d: (\{0, 1\} \rightarrow n-1)$ ,  $e: (Q_n \setminus \{0\} \rightarrow n-1)(0 \rightarrow 1)$ , and  $F = \{q \in Q_n \setminus \{0, n-1\} \mid q \text{ is odd}\}$ . For  $n = 4$ ,  $a$  and  $b$  coincide, and we can use  $\Sigma = \{b, c, d, e\}$ .*

*Let  $L_n(a, b, c, d, e)$  be the language of  $\mathcal{D}_n(a, b, c, d, e)$ .*

**Theorem 13 (Semigroup, Quotients, Reversal, Atoms, Boolean Ops).**  $L_n(a, b, c, d, e)$  is a suffix-free language of complexity  $n$ . Moreover, it meets the following bounds: 1. Semigroup size:  $(n-1)^{n-2} + n - 2$  for  $n \geq 6$ . 2. Quotient complexities:  $n - 1$ , except  $\kappa(L) = n$ ,  $\kappa(\emptyset) = 1$ . 3. Reversal:  $2^{n-2} + 1$ . 4. Atom complexities:  $\kappa(A_S) = 2^{n-2} + 1$ , if  $S = \emptyset$ ;  $\kappa(A_S) = n$ , if  $S = \{0\}$ ;  $\kappa(A_S) = 1 + \sum_{x=1}^{|S|} \sum_{y=0}^{n-2-|S|} \binom{n-2}{x} \binom{n-2-x}{y}$ , if  $\emptyset \neq S \subseteq \{1, \dots, n-2\}$ . 5. Boolean operations:  $mn - (m+n-2)$  if  $\circ \in \{\cup, \oplus\}$ ,  $mn - (m+2n-4)$  if  $\circ = \setminus$ , and  $mn - 2(m+n-3)$  if  $\circ = \cap$ .

**Definition 10.** For  $n \geq 4$ , define the DFA  $\mathcal{D}_n(a, b, c) = (Q_n, \Sigma, \delta, 0, \{n-2\})$ , where  $Q_n = \{0, \dots, n-1\}$ ,  $\Sigma = \{a, b, c\}$ , and  $\delta$  is defined by  $a: (0 \rightarrow n-1)(1, \dots, n-2)$ ,  $b: (0 \rightarrow n-1)(1, 2)$ ,  $c: (1, n-1)(0 \rightarrow 1)$ . Let  $L_n(a, b, c)$  be the language of  $\mathcal{D}_n(a, b, c)$ .

**Theorem 14 (Star, Product, Boolean Operations).**  $L_n(a, b, c)$  and its dialects meet the bounds for star, product, and boolean operations as follows: 1. Star:  $2^{n-2} + 1$ . 2. Product:  $(n-1)2^{n-2} + 1$ . 3. Boolean operations: as in Theorem 13.

## BIFIX-FREE LANGUAGES

A language is *bifix-free* if it is both prefix-free and suffix-free. The complexities of common operations using various witnesses were studied in [16], a conjecture on the semigroup size in [20], and tight upper bound in [60]. Since every bifix-free language has an empty quotient, the restricted and unrestricted cases for binary operations coincide. The results below were found recently [41].

**Definition 11.** For  $n \geq 7$ , define the DFA  $\mathcal{D}_n(a, b, c) = (Q_n, \Sigma, \delta, 0, \{n-2\})$ , where  $Q_n = \{0, \dots, n-1\}$ ,  $\Sigma = \{a, b, c\}$ ,  $h = \lfloor (n-1)/2 \rfloor$ ,  $\delta$  is given by  $a: (0 \rightarrow 1)(\{1, \dots, n-3\} \rightarrow n-2)(\{n-2, n-1\} \rightarrow n-1)$ ,  $b: (\{0, n-2, n-1\} \rightarrow n-1)(1, \dots, n-3)$ , and  $c: (\{0, n-2, n-1\} \rightarrow n-1)(1 \rightarrow h)(h \rightarrow n-2)(n-3, \dots, h+1, h-1, \dots, 2)$ . Let  $L_n(a, b, c)$  be the language of  $\mathcal{D}_n(a, b, c)$ .

**Theorem 15 (Bounds for Operations).** The DFA of Definition 11 is minimal and its language  $L_n(a, b, c)$  is bifix-free and has complexity  $n$ . The stream  $(L_n(a, b, c) \mid n \geq 9)$  with some dialect streams meets the bounds for common operations on bifix-free languages: 1. Quotient complexities:  $n - 1$ , except  $\kappa(L) = n$ ,  $\kappa(\{\varepsilon\}) = 2$ , and  $\kappa(\emptyset) = 1$ . 2. Reversal:  $2^{n-3} + 2$ . 3. Star:  $n - 1$ . 4. Product:  $m+n-2$ . 5. Boolean operations:  $mn - (m+n)$  if  $\circ \in \{\cup, \oplus\}$ ,  $mn - (2m+3n-9)$  if  $\circ = \setminus$ , and  $mn - 3(m+n-4)$  if  $\circ = \cap$ .

Even though bifix-free languages are a subclass of suffix-free languages and there does not exist a most complex suffix-free stream, we do have a most complex bifix-free stream. This stream has an alphabet of size  $(n-2)^{n-3} + (n-3)2^{n-3} - 1$  [41], and the alphabet size cannot be reduced. The syntactic semigroup of this language is of size  $(n-1)^{n-3} + (n-2)^{n-3} + (n-3)2^{n-3}$ . Maximal atom complexities are:  $\kappa(A_S) = 2^{n-2} + 1$ , if  $S = \emptyset$ ;  $\kappa(A_S) = n$ , if  $S = \{0\}$ ;  $\kappa(A_S) = 2$ , if  $S = \{n-2\}$ ;  $\kappa(A_S) = 3 + \sum_{x=1}^{|S|} \sum_{y=0}^{n-3-|S|} \binom{n-3}{x} \binom{n-3-x}{y}$ , if  $\emptyset \neq S \subseteq \{1, \dots, n-3\}$ .

For further details see [41,60].

**PROPER SUFFIX-CONVEX LANGUAGES** This is the second class found for which a most complex stream does not exist. The complexity of this class is still being studied, but we do know that at least three different witnesses are required to meet the bounds for all the measures<sup>4</sup>.

### 9.3 Factor-Convex Languages

**TWO-SIDED IDEALS** The complexities of basic operations on two-sided ideals were studied in [15]. The following stream of two-sided ideals was defined in [35], where it was conjectured that the DFAs in this stream have maximal transition semigroups. This was proved in [28], and the stream was shown to be most complex for restricted operations in [14]. It is also most complex in the unrestricted case [24].

**Definition 12.** For  $n \geq 5$ , let  $\mathcal{D}_n = \mathcal{D}_n(a, b, c, d, e, f) = (Q_n, \Sigma, \delta, 0, \{n-1\})$ , where  $\Sigma = \{a, b, c, d, e, f\}$ , and  $\delta_n$  is defined by  $a: (1, 2, \dots, n-2)$ ,  $b: (1, 2)$ ,  $c: (n-2 \rightarrow 1)$ ,  $d: (n-2 \rightarrow 0)$ ,  $e: (Q_{n-1} \rightarrow 1)$ , and  $f: (1 \rightarrow n-1)$ . Let  $L_n(a, b, c, d, e, f)$  be the language of  $\mathcal{D}_n(a, b, c, d, e, f)$ .

**Theorem 16 ( Most Complex Two-Sided Ideals).** For  $n \geq 5$ , the language  $L_n(a, b, c, d, e, f)$  is a two-sided ideal of complexity  $n$ . The witness stream  $(L_n(a, b, c, d, e, f) \mid n \geq 5)$  with some dialect streams is most complex in the class of regular two-sided ideals, and meets the following complexity bounds: 1. Semigroup size:  $n^{n-2} + (n-2)2^{n-2} + 1$ . 2. Quotient complexities:  $n$ . 3. Reversal:  $2^{n-1} + 1$ . 4. Atom complexities:  $\kappa(A_S) = n$ , if  $S = Q_n$ ;  $\kappa(A_S) = 2^{n-2} + n - 1$ , if  $S = Q_n \setminus \{1\}$ ;  $\kappa(A_S) = 1 + \sum_{x=1}^{|S|} \sum_{y=1}^{n-|S|} \binom{n-2}{x-1} \binom{n-x-1}{y-1}$ , otherwise. 5. Star:  $n + 1$ . 6. (a) Restricted product:  $m + n - 1$ ; (b) unrestricted product:  $m + 2n$ . 7. (a) Restricted boolean operations:  $mn$  if  $\circ \in \{\cap, \oplus\}$ ,  $mn - (m - 1)$  if  $\circ = \setminus$ ,  $mn - (m + n - 2)$  if  $\circ = \cup$ . (b) unrestricted boolean operations: same as regular languages. At least six letters are required to meet all these bounds [24].

**FACTOR-CLOSED LANGUAGES** The complexities of basic operations on factor-closed languages were examined in [17]. The syntactic complexity of factor-closed languages is the same as that of two-sided ideals, because each factor-closed language other than  $\Sigma^*$  is the complement of a two-sided ideal. Most complex factor-closed languages have not been studied.

**FACTOR-FREE LANGUAGES** The complexities of basic operations on factor-free languages were examined in [16]. The syntactic complexity of factor-free languages was conjectured in [20] to be  $(n-1)^{n-3} + (n-3)2^{n-3} + 1$ , but the problem is still open. Most complex factor-free languages have not been studied.

---

<sup>4</sup> C. Sinnamon: private communication

### 9.4 Subword-Convex Languages

**ALL-SIDED IDEALS** The complexities of basic operations were examined in [15]. The syntactic complexity has not been studied.

**SUBWORD-CLOSED LANGUAGES** The complexities of basic operations were examined in [17]. The syntactic complexity has not been studied.

**SUBWORD-FREE LANGUAGES** The complexities of basic operations were examined in [16]. The syntactic complexity has not been studied.

### 9.5 Other Classes

#### NON-RETURNING LANGUAGES

A deterministic finite automaton (DFA) is *non-returning* if there are no transitions into its initial state. A regular language is non-returning if its minimal DFA has that property. The state complexities of common operations (boolean operations, Kleene star, reverse and product) were studied by Eom, Han and Jirásková [40]. Most complex non-returning languages were examined in [13].

If  $t$  has rank  $n - 1$ , there is exactly one pair of distinct elements  $i, j \in Q_n$  such that  $it = jt$ . A transformation  $t$  of  $Q_n$  is of *type*  $\{i, j\}$  if  $t$  has rank  $n - 1$  and  $it = jt$  for  $i < j$ .

Let  $\Gamma = \{a_{i,j} \mid 0 \leq i < j \leq n - 1\}$ , where  $a_{i,j}$  is a letter that induces any transformation of type  $\{i, j\}$  and does not map any state to 0. Let  $\Gamma' = \Gamma \setminus \{a_{0,n-1}, a_{0,1}, a_{1,n-1}, a_{0,2}\}$ . Let  $\Sigma = \{a, b, c, d\} \cup \Gamma'$ , where  $a : (1, \dots, n - 1)(0 \rightarrow 1)$ ,  $b : (1, 2)(0 \rightarrow 2)$ ,  $c : (2, \dots, n - 1)(1 \rightarrow 2)(0 \rightarrow 1)$ , and  $d : (0 \rightarrow 2)$ .

**Definition 13.** For  $n \geq 4$ , let  $\mathcal{D}_n = \mathcal{D}_n(\Sigma) = (Q_n, \Sigma, \delta_n, 0, \{n - 1\})$ , where  $\Sigma = \{a, b, c, d\} \cup \Gamma'$ , and  $\delta_n$  is defined in accordance with the transformations described above. Let  $L_n = L_n(\Sigma)$  be the language accepted by  $\mathcal{D}_n(\Sigma)$ .

**Theorem 17 (Most Complex Non-Returning Languages).** For each  $n \geq 4$ , the DFA of Definition 13 is minimal and non-returning. The stream  $(L_n(\Sigma) \mid n \geq 4)$  with some dialect streams is most complex in the class of regular non-returning languages and meets the bounds: 1. Semigroup size:  $(n - 1)^n$ . 2. Quotient complexities:  $n - 1$ , except  $\kappa(L) = n$ . 3. Reversal:  $2^n$ . 4. Atom complexities:  $\kappa(A_S) = 2^{n-1}$ , if  $S \in \{\emptyset, Q_n\}$ ;  $\kappa(A_S) = 2 + \sum_{x=1}^{|S|} \sum_{y=1}^{|S|} \binom{n-1}{x} \binom{n-1-x}{y}$ , otherwise. 5. Star:  $2^{n-1}$ . 6. (a) Restricted product:  $(m - 1)2^{n-1} + 1$ ; (b) unrestricted product:  $m2^{n-1} + 1$ . 7. (a) Restricted boolean operations:  $mn - (m + n - 2)$ ; (b) unrestricted boolean operations:  $(m + 1)(n + 1)$  if  $\circ \in \{\cup, \oplus\}$ ,  $mn + m$  if  $\circ = \setminus$ ,  $mn$  if  $\circ = \cap$ .

The bound on the semigroup size and on the complexity of atoms require an alphabet of at least  $\binom{n}{2}$  letters, reversal requires at least three letters, and all the other bounds can be met by binary witnesses.

## STAR-FREE LANGUAGES

A language is *star-free* if it can be constructed from the basic languages using only boolean operations and product, but no star. A famous theorem by Schützenberger [58] states that a language is star-free if and only if its syntactic monoid is aperiodic, meaning that it contains only trivial one-element groups. Star-free languages have many interesting subclasses [7].

The complexities of basic operations on star-free languages were studied in [21]. It is surprising that these languages can meet all the bounds for regular languages, except for reversal, which has a tight upper bound of  $2^n - 1$  [27]. Most complex star-free languages have not been studied mainly because no tight upper bound on their syntactic complexity is known, even though some large aperiodic semigroups have been found [29].

Syntactic complexities for several subclasses of star-free languages have been found:

1. A language is *J-trivial* if its syntactic monoid  $M$  satisfies the following:  $MsM = MtM$  implies  $s = t$ , for all  $s, t \in M$ . It has been shown in [18] that the syntactic complexity of *J-trivial* languages is  $\lfloor e(n-1)! \rfloor$ .
2. A language is *R-trivial* if its syntactic monoid  $M$  satisfies the following:  $sM = tM$  implies  $s = t$ , for all  $s, t \in M$ . The syntactic complexity of *R-trivial* languages is  $n!$  [18].
3. A language is *cofinite* if its complement is finite. The syntactic complexity of the class of finite and cofinite languages is  $(n-1)!$  [19].
4. A language is *reverse definite* if it can be expressed in the form  $L = E \cup F\Sigma^*$ , where  $E$  and  $F$  are finite. The syntactic complexity of reverse definite languages is  $(n-1)!$  [19].

## 10 Groups and Complexity

We close this paper with a brief mention of some group-theoretic results that simplify certain proofs about complexity.

Let  $S_n$  denote the symmetric group of degree  $n$ . A *basis* of  $S_n$  is an ordered pair  $(s, t)$  of distinct transformations of  $Q_n = \{0, \dots, n-1\}$  that generate  $S_n$ . Two bases  $(s, t)$  and  $(s', t')$  of  $S_n$  are *conjugate* if there exists a transformation  $r \in S_n$  such that  $rsr^{-1} = s'$ , and  $rtr^{-1} = t'$ .

Assume that a DFA  $\mathcal{D}'_m$  (respectively,  $\mathcal{D}_n$ ) has state set  $Q'_m$  ( $Q_n$ ), and let the subgroup of permutations of its transition semigroup be  $S'_m$  ( $S_n$ ). Let  $L'_m$  ( $L_n$ ) be the language accepted by  $\mathcal{D}'_m$  ( $\mathcal{D}_n$ ). The following was proved in [3]:

**Theorem 18.** *Suppose  $m, n \geq 2$  and  $(m, n) \notin \{(2, 2), (3, 4), (4, 3), (4, 4)\}$ . If the subgroups of permutations in the transition semigroups of  $\mathcal{D}'_m$  and  $\mathcal{D}_n$  are  $S'_m$  and  $S_n$  respectively, and  $\circ$  is a proper binary boolean operation, then the complexity of  $L'_m \circ L_n$  is  $mn$ , unless  $m = n$  and the bases induced by the letters of  $\Sigma$  in the transition semigroups of  $\mathcal{D}'_m$  and  $\mathcal{D}_n$  are conjugate, in which case the quotient complexity of  $L'_m \circ L_n$  is at most  $m = n$ .*

In the DFAs used for most complex streams, usually the transition semigroups contain all permutations of some subset of the state set. Theorem 18 has greatly simplified the proofs of results about the complexity of boolean operations in several cases [13,14,24,26,28].

In the special case where  $\mathcal{D}'_m$  and  $\mathcal{D}_n$  are DFAs with exactly one final state, which occurs very commonly in most complex streams, there is a stronger result due to Davies [37].

Recall that to recognize boolean operations on the languages of  $\mathcal{D}'_m$  and  $\mathcal{D}_n$ , we use the direct product DFA  $\mathcal{D}'_m \times \mathcal{D}_n$  with state set  $Q'_m \times Q_n$ . A *row* of  $Q'_m \times Q_n$  is a set of the form  $R_{p'} = \{(p', q) : q \in Q_n\}$ . A *column* of  $Q'_m \times Q_n$  is a set of the form  $C_q = \{(p', q) : p' \in Q'_m\}$ .

We say a state  $q$  of a DFA  $\mathcal{D}$  is *reachable by permutations* if it is reachable by some word  $w$  that induces a permutation in the transition semigroup of  $\mathcal{D}$ . If the transition semigroup of  $\mathcal{D}$  is a group, this is the same thing as just being reachable.

**Theorem 19.** *Suppose  $m, n \geq 2$  and  $(m, n) \neq (2, 2)$ . Let  $\mathcal{D}'_m$  and  $\mathcal{D}_n$  be minimal DFAs with  $m$  and  $n$  states respectively. Suppose that every state in each DFA is reachable by permutations, and each DFA has exactly one final state. Consider the direct product  $\mathcal{D}'_m \times \mathcal{D}_n$ . The following are equivalent:*

1. *Every state in  $Q'_m \times Q_n$  is reachable by permutations.*
2. *There exists  $p' \in Q'_m$  such that every state in row  $R_{p'} \subseteq Q'_m \times Q_n$  is reachable by permutations.*
3. *There exists  $q \in Q_n$  such that every state in column  $C_q \subseteq Q'_m \times Q_n$  is reachable by permutations.*
4. *The complexity of  $L'_m \circ L_n$  is  $mn$  for all proper binary boolean operations  $\circ$ .*

It was proved in [3] that if  $\mathcal{D}'_m$  and  $\mathcal{D}_n$  satisfy the conditions of Theorem 18, then every state in  $Q'_m \times Q_n$  is reachable by permutations. However, this result applies more generally, including in cases where the transition semigroups of  $\mathcal{D}'_m$  and  $\mathcal{D}_n$  are not symmetric groups. The downside is the restriction on the final state sets.

The next result<sup>5</sup> greatly simplified a proof about product [24]. Suppose  $\mathcal{D}'_m = (Q'_m, \Sigma, \delta', 0', \{f'\})$  is a minimal DFA of  $L'_m$ ,  $f' \neq 0'$ , and  $\mathcal{D}_n = (Q_n, \Sigma, \delta, 0, F)$  is a minimal DFA of  $L_n$ . We use the normal construction of an  $\varepsilon$ -NFA  $\mathcal{N}$  – an NFA that permits also transitions induced by the empty word – to recognize  $L'_m L_n$ , by introducing an  $\varepsilon$ -transition from the final state of  $\mathcal{D}'_m$  to the initial state of  $\mathcal{D}_n$ , and changing the final state of  $\mathcal{D}'_m$  to non-final. We need to show that the following types of sets are reachable from the initial set  $\{0'\}$  in the subset construction for  $\mathcal{N}$ : (a)  $(m-1)2^n$  sets  $\{p'\} \cup S$ , where  $p' \in Q'_m \setminus \{f'\}$ , and  $S \subseteq Q_n$ , (b)  $2^{n-1}$  sets  $\{f', 0\} \cup S$ , where  $S \subseteq Q_n \setminus \{0\}$ . The lemma below allows us to check the reachability of only a few special sets.

<sup>5</sup> S. Davies: private communication

**Lemma 1.** *If the transition semigroups of  $\mathcal{D}'_m$  and  $\mathcal{D}_n$  are groups, and all the sets of the form  $\{p'\}$ ,  $p' \in Q'_m \setminus \{f'\}$ , and  $\{0', q\}$ ,  $q \in Q_n$  are reachable, then so are all sets of the form*

$$\{p'\} \cup S, p' \in Q'_m \setminus \{f'\}, S \subseteq Q_n \text{ and } \{f', 0\} \cup S, S \subseteq Q_n \setminus \{0\}.$$

## 11 Conclusions

We have surveyed many papers concerned with complexity measures for regular languages and finite automata, and put special emphasis on most complex languages because they concisely describe the properties of the given languages. However, some questions remain.

Upper bounds on syntactic complexity are known for several subclasses. Finding these upper bounds was trivial for right ideals and non-returning languages, easy for prefix-free and proper prefix-convex languages, and challenging for left ideals and two-sided ideals, suffix-free and bifix-free languages. The problem remains open for factor-free, and subword-free languages, and all-sided ideals. As well, this question is open for star-free languages and many proper subclasses of star-free languages [7], for example, definite languages [19] and  $L$ -trivial languages, where a language is  $L$ -trivial if its syntactic monoid  $M$  satisfies the following:  $Ms = Mt$  implies  $s = t$ , for all  $s, t \in M$ . Of course, if the syntactic complexity is unknown, then so is the existence of most complex language streams.

We have included atom complexities as a measure, but more work needs to be done to determine their usefulness. We justify the inclusion of atom complexities by the following observation: so far, whenever a language stream meets the bounds for the basic operations and syntactic complexity, it also meets the bounds for atom complexities.

Finally, it would be very useful to have more results like those in Section 10 because they allow us to avoid complex proofs or greatly simplify them.

**Acknowledgment** I am very grateful to Sylvie Davies, Corwin Sinnamon, Marek Szykuła, and Hellis Tamm not only for careful proofreading, but also for their many contributions to this paper.

## References

1. Ang, T., Brzozowski, J.A.: Languages convex with respect to binary relations, and their closure properties. *Acta Cybernet.* 19(2), 445–464 (2009)
2. Beaumont, R.A., Peterson, R.P.: Set-transitive permutation groups. *Canadian Journal of Mathematics* 7, 35–42 (1955)
3. Bell, J., Brzozowski, J.A., Moreira, N., Reis, R.: Symmetric groups and quotient complexity of boolean operations. In: Esparza, J., et al. (eds.) *ICALP 2014*. LNCS, vol. 8573, pp. 1–12. Springer (2014)
4. Berstel, J., Perrin, D., Reutenauer, C.: *Codes and Automata*. Cambridge University Press (2010)



5. Brzozowski, J.A.: Canonical regular expressions and minimal state graphs for definite events. In: Fox, J. (ed.) *Mathematical Theory of Automata*, pp. 529–561. MRI Symposia Series, vol. 12, Polytechnic Press of the Polytechnic Institute of Brooklyn (1963)
6. Brzozowski, J.A.: Derivatives of regular expressions. *J. ACM* 11(4), 481–494 (1964)
7. Brzozowski, J.A.: Hierarchies of aperiodic languages. *R.A.I.R.O.-Informatique théorique* 10(R2), 33–49 (1976)
8. Brzozowski, J.A.: In search of the most complex regular languages. *Int. J. Found. Comput. Sc.* 24(6), 691–708 (2013)
9. Brzozowski, J.A.: Towards a theory of complexity for regular languages. In: Brlek, S., Reutenauer, C. (eds.) *DLT 2016*. LNCS, vol. 9840, p. XI. Springer (2016)
10. Brzozowski, J.A.: Unrestricted state complexity of binary operations on regular languages. In: C. Câmpeanu et al. (ed.) *DCFS*. LNCS, vol. 9777, pp. 60–72. Springer (2016)
11. Brzozowski, J.A., Davies, G.: Maximally atomic languages. In: Ésik, Z., Fülöp, Z. (eds.) *Automata and Formal Languages (AFL 2014)*. pp. 151–161. EPTCS (2014)
12. Brzozowski, J.A., Davies, S.: Quotient complexities of atoms of regular ideal languages. *Acta Cybernet.* 22, 293–311 (2015)
13. Brzozowski, J.A., Davies, S.: Most complex non-returning regular languages (2017), <http://arxiv.org/abs/1701.03944>
14. Brzozowski, J.A., Davies, S., Liu, B.Y.V.: Most complex regular ideal languages. *Discrete Math. Theoret. Comput. Sc.* 18(3) (2016), paper #15
15. Brzozowski, J.A., Jirásková, G., Li, B.: Quotient complexity of ideal languages. *Theoret. Comput. Sci.* 470, 36–52 (2013)
16. Brzozowski, J.A., Jirásková, G., Li, B., Smith, J.: Quotient complexity of bifix-, factor-, and subword-free regular languages. *Acta Cybernet.* 21(4), 507–527 (2014)
17. Brzozowski, J.A., Jirásková, G., Zou, C.: Quotient complexity of closed languages. *Theory Comput. Syst.* 54, 277–292 (2014)
18. Brzozowski, J.A., Li, B.: Syntactic complexity of  $\mathcal{R}$ - and  $\mathcal{J}$ -trivial languages. *Internat. J. Found. Comput. Sci.* 25(7), 807–821 (2014)
19. Brzozowski, J.A., Li, B., Liu, D.: Syntactic complexities of six classes of star-free languages. *J. Autom. Lang. Comb.* 17(2–4), 83–105 (2012)
20. Brzozowski, J.A., Li, B., Ye, Y.: Syntactic complexity of prefix-, suffix-, bifix-, and factor-free regular languages. *Theoret. Comput. Sci.* 449, 37 – 53 (2012)
21. Brzozowski, J.A., Liu, B.: Quotient complexity of star-free languages. *Internat. J. Found. Comput. Sci.* 23(6), 1261–1276 (2012)
22. Brzozowski, J.A., Shallit, J., Xu, Z.: Decision problems for convex languages. *Inform. and Comput.* 209, 353–367 (2011)
23. Brzozowski, J.A., Sinnamon, C.: Complexity of prefix-convex regular languages (2016), <http://arxiv.org/abs/1605.06697>
24. Brzozowski, J.A., Sinnamon, C.: Unrestricted state complexity of binary operations on regular and ideal languages (2016), <http://arxiv.org/abs/1609.04439>
25. Brzozowski, J.A., Sinnamon, C.: Complexity of left-ideal, suffix-closed and suffix-free regular languages. In: *LATA*. LNCS, Springer (2017), to appear
26. Brzozowski, J.A., Sinnamon, C.: Complexity of right-ideal, prefix-closed, and prefix-free regular languages. *Acta Cybernet.* (2017), to appear
27. Brzozowski, J.A., Szykuła, M.: Large aperiodic semigroups, <http://arxiv.org/abs/11401.0157>
28. Brzozowski, J.A., Szykuła, M.: Upper bounds on syntactic complexity of left and two-sided ideals. In: Shur, A.M., Volkov, M.V. (eds.) *DLT 2014*. LNCS, vol. 8633, pp. 13–24. Springer (2014)

29. Brzozowski, J.A., Szykuła, M.: Large aperiodic semigroups. *Int. J. Found. Comput. Sc.* 26(7), 913–931 (2015)
30. Brzozowski, J.A., Szykuła, M.: Complexity of suffix-free regular languages. *J. Comput. System Sci.* (2017), to appear. Also at <http://arxiv.org/abs/1504.05159>
31. Brzozowski, J.A., Szykuła, M.: Syntactic complexity of suffix-free languages. *Inform. and Comput.* (2017), to appear. Also at <http://arxiv.org/abs/1412.2281>
32. Brzozowski, J.A., Szykuła, M., Ye, Y.: Syntactic complexity of regular ideals (2015), <http://arxiv.org/abs/1509.06032>
33. Brzozowski, J.A., Tamm, H.: Complexity of atoms of regular languages. *Int. J. Found. Comput. Sc.* 24(7), 1009–1027 (2013)
34. Brzozowski, J.A., Tamm, H.: Theory of átomata. *Theoret. Comput. Sci.* 539, 13–27 (2014)
35. Brzozowski, J.A., Ye, Y.: Syntactic complexity of ideal and closed languages. In: Mauri, G., Leporati, A. (eds.) *DLT. LNCS*, vol. 6795, pp. 117–128. Springer (2011)
36. Čmórik, R., Jirásková, G.: Basic operations on binary suffix-free languages. In: Kotásek, Z., et al. (eds.) *MEMICS*. pp. 94–102 (2012)
37. Davies, S.: Primitivity, minimality and state complexity of boolean operations (2017), <http://arxiv.org/abs/1702.00877>
38. Diekert, V., Walter, T.: Asymptotic approximation for the quotient complexities of atoms. *Acta Cybernetica* 22(2), 349–357 (2015)
39. Eilenberg, S.: *Automata, Languages, and Machines*, vol. Vol. B. Academic Press (1976)
40. Eom, H.S., Han, Y.S., Jirásková, G.: State complexity of basic operations on non-returning regular languages. *Fund. Inform.* 144, 161–182 (2016)
41. Ferens, R., Szykuła, M.: Complexity of regular bifix-free languages (2016), <http://arxiv.org/abs/1604.06936>
42. Han, Y.S., Salomaa, K.: State complexity of union and intersection of finite languages. *Int. J. Found. Comput. Sci.* 19(3), 581–595 (2008)
43. Han, Y.S., Salomaa, K., Wood, D.: Operational state complexity of prefix-free regular languages. In: Ésik, Z., Fülöp, Z. (eds.) *Automata, Formal Languages, and Related Topics*. pp. 99–115. Institute of Informatics, University of Szeged, Hungary (2009)
44. Han, Y.S., Salomaa, K.: State complexity of basic operations on suffix-free regular languages. *Theoret. Comput. Sci.* 410(27–29), 2537–2548 (2009)
45. Hopcroft, J.E.: An  $n \log n$  algorithm for minimizing states in a finite automaton. In: Kohavi, Z., Paz, A. (eds.) *Theory of Machines and Computations*, pp. 189–196. Academic Press (1971)
46. Iván, S.: Complexity of atoms, combinatorially. *Inform. Process. Lett.* 116(5), 356–360 (2016)
47. Jirásková, G., Krausová, M.: Complexity in prefix-free regular languages. In: McQuillan, I., Pighizzini, G., Trost, B. (eds.) *Proceedings of the 12th International Workshop on Descriptive Complexity of Formal Systems (DCFS)*. pp. 236–244. University of Saskatchewan (2010)
48. Jirásková, G., Olejár, P.: State complexity of union and intersection of binary suffix-free languages. In: Bordihn, H., et al. (eds.) *NMCA*. pp. 151–166. Austrian Computer Society (2009)
49. Kleene, S.C.: Representation of events in nerve nets and finite automata. In: Shannon, C.E., McCarthy, J. (eds.) *Automata Studies*, pp. 3–41. *Annals of Mathematics Studies*, no. 34, Princeton University Press (1956)
50. Krausová, M.: Prefix-free regular languages: Closure properties, difference, and left quotient. In: *MEMICS*. pp. 114–122 (2011)

51. Maslov, A.N.: Estimates of the number of states of finite automata. Dokl. Akad. Nauk SSSR 194, 1266–1268 (Russian). (1970), English translation: Soviet Math. Dokl. **11** (1970) 1373–1375
52. Moore, E.F.: Gedanken experiments on sequential machines. In: Shannon, C.E., McCarthy, J. (eds.) Automata Studies, pp. 129–153. Annals of Mathematics Studies, no. 34, Princeton University Press (1956)
53. Myhill, J.: Finite automata and representation of events. Wright Air Development Center Technical Report 57–624 (1957)
54. Nerode, A.: Linear automaton transformations. Proc. Amer. Math. Soc. 9, 541–544 (1958)
55. Perrin, D.: Finite automata. In: van Leeuwen, J. (ed.) Handbook of Theoretical Computer Science, vol. B, pp. 1–57. Elsevier (1990)
56. Pin, J.E.: Syntactic semigroups. In: Handbook of Formal Languages, vol. 1: Word, Language, Grammar, pp. 679–746. Springer, New York, NY, USA (1997)
57. Salomaa, A., Wood, D., Yu, S.: On the state complexity of reversals of regular languages. Theoret. Comput. Sci. 320, 315–329 (2004)
58. Schützenberger, M.: On finite monoids having only trivial subgroups. Inform. and Control 8, 190–194 (1965)
59. Shyr, H.J., Thierrin, G.: Hypercodes. Inform. and Control 24, 45–54 (1974)
60. Szykuła, M., Wittnebel, J.: Syntactic complexity of bifix-free languages (2016), <http://arxiv.org/abs/1604.06936>
61. Thierrin, G.: Convex languages. In: Nivat, M. (ed.) Automata, Languages and Programming, pp. 481–492. North-Holland (1973)
62. Yu, S.: Regular languages. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, pp. 41–110. Springer (1997)
63. Yu, S., Zhuang, Q., Salomaa, K.: The state complexities of some basic operations on regular languages. Theoret. Comput. Sci. 125, 315–328 (1994)